TAIG

Jan. 1982

Chairman Steve Crowley 937-1001 Tr Jim Dahlberg 432-1963
Sec Mike Doleman 861-1893

26 LINE DISPLAY!

By Jim Dahlberg

The following program can display 26 text lines on your TV if you have at least 40k of memory and a good TV! If you don't have 40k of memory it is perfectly feasible to relocate the display list POKES to addresses which are available on any system. It doesn't matter at all where you relocate the display list to.

When you run the program it is supposed to rewrite the display list and then write some data to the new screen memory area to display some line numbers on your screen. I say "supposed to" because it doesn't always work exactly right the first time. If you get a blank screen or any other kind of wild display, just hit SYSTEM RESET and run it again.

To understand how the program works read the REM statements in the listing. Basically it just writes a new display list, and informs the operating system (OS) where the new display list is located and where the new address for the display data is. The remainder of the POKES write data to the display area directly. The reason that PRINT statements could not be used is that OS does not know that there are two extra text lines on your TV, therefor you must POKE the data directly into memory. One other complication is that you cannot poke ATASCII data into memory. You must use the internal character codes. (No one said it would be easy)

```
5 POKE 39999,112:REM START OF DISPLAY LIST; 8 BLANK LINES
10 POKE 40000,112:POKE 40001,66:POKE 40002,144:POKE 40003,151:REM 8 MORE BLANK LINES;SET MEMORY SCAN TO 38800
20 FOR I=40004 TO 40028:POKE I,2:NEXT I:REM 25 MORE GR.0 MODE LINES
30 POKE 40030,65:POKE 40031,64:POKE 40032,156:REM JUMP BACK TO START OF DISPLAY LIST
40 POKE 560,63:POKE 561,156:REM START NEW DISPLAY LIST
50 POKE 88,144:POKE 89,151:REM INFORM OS OF NEW LOCATION FOR DISPLAY DATA
60 POKE 39760,44:POKE 39761,41:POKE 39762,46:POKE 39763,37:POKE 39764,0:POKE 39765,18:POKE 39766,21
65 REM THE FOLLOWING POKES WRITE SOME DATA INTO THE NEW DISPLAY AREA TO
66 REM SHOW THAT THERE ARE 26 LINES DISPLAYED.
70 POKE 39800,44:POKE 39801,41:POKE 39802,46:POKE 39803,37:POKE 39804,0:POKE 39805,18:POKE 39806,22
80 POKE 38800,44:POKE 38801,41:POKE 38802,46:POKE 38803,37:POKE 38804,0:POKE 38805,17
90 STOP :REM THAT'S IT FOLKS!
100 FOR I=38800 TO 39880:POKE I,0:NEXT I:REM THIS IS JUST A ROUTINE TO CLEAR THE NEW DISPLAY AREA
```

Since this program was designed only to test the possibility of displaying extra lines no attempt was made to make it a "clean" program. Yes, there are other problems with it. If you try and clear the screen with the ATARI clear code, the screen will blank out. Also if you list a program or do anything else that will cause the screen to go beyond the normal 24 lines of display, the display will go haywire. To clear the screen just run the program and when it stops type GOTO 100. Line 100 is just a routine to clear the display memory.

Possible uses for this technique (properly cleaned up of course) are extra lines displaying some kind of status information while a program is making normal use of the rest of the screen.

ASM PRINT UPDATE

By Mike Doleman

Several months ago I wrote several articles on a method to do a PRINT in assembly language. Now Atari is apparently changing the addresses of some of thier routines in thier new version operating systems, a concequence of which programs which directly call these routines at thier old OS addresses fail to work. The point is that Atari never intended these routines to be called directly, and instead provided a method called vectoring to call these routines no matter where they are in the OS. The problem is that the principle of doing this involves a little not so obvious coding.

A vector is simply an address in RAM thet contains the address you want. This means that although Atari may move the routines around in the OS they will always tell you where they moved them in the vector address.

The only thing you have to know is where the vector address is, and the location of this vector is what must never change. It turns out that the vector address for the routine for printing a character on the screen is at $E406 (a hex $), in the old OS the address at $E406 (& $E407) is $F6A3. But wait! The address we jumped to last time was $F6A4! Here is where the not so obvoius coding is needed.

All the vector addresses are the address you want MINUS one. This means if you do an indirect JMP you end up in the wrong place and probably get messed up. I won't try to explain why the vector addresses are all minus one, since I don't know (it seems that they took a simple task and complicated it). What I will do is show how to use it.

Rather than doing an indirect JMP to the routine you must first put the address you want to go to on the stack using PHA. Then you do a RTS (Return from subroutine). This works due to the nature of the RTS, which allways adds one to the address it finds on the stack.

This is what the code should look like:

```
100  LDA $E407
110  PHA          be sure second byte goes on stack first.
120  LDA $E406
130  PHA          then put first byte on.
140  LDA #XX      XX= ASCII code for letter you want printed.
150  RTS          go to subroutine.
```

But wait! We still are headed for trouble when the subroutine we just so cleverly "jumped" to does its own RTS. So we must also put code in to compensate for that, and the best way I can see is to make the above a subroutine itself to avoid any more sillyness of LDAing & PHAing the stack.

If we call the above routine PRINT, then:

```
30  LDA #XX
40  STA $02FB    OS uses $02FB for this so we will to.
50  JSR PRINT    go to our subroutine.
```

One more thing to do, pant pant, and we'll be set. Change the #XX at line 140 in the subroutine to $02FB.

I think life would have been simpler if the right address would have been put in the vector.

SPRICHTS SIE, ATARI!

I did it again, I managed to think up an artical of relavence. (This one will be a little hurried, I must find a way home from Computer Castle, for you people in the more baumy lattitudes, we die hard Minnesotans are braving another typical winter-tonight 18 inches+ of snow.) I bet all of you T.A.I.G. members and you prospective members who read this prestigous periodical remember my question raised (I think it was last month) when I reviewed an artical dealing with Apple voice synthesis. (The exact question was can one read the value of lets say, a song, through the cassette. The advantage of this is on can easily digitilize music, voices, etc.. without having to write a program to sample a pot while speaking over a mic, whew! Oh by the way, since this newsletter is starting to get are>ound, I like to give a hint to Mike Dunn of A.C.E.-character is not spelled charactor. During the past few months I've noticed this solecisism quite frequently)
Okay, what am I talking about. In De Re Atari, in the chapter dealing with sound, they mention that it is possible to make the Atari say 'hello'. Thats true (I'm pretty sure anyhow). The easiest way to do this is to hook a run of the mill microphone to a small amplifier (lets say from a stero's phono output and then use the speaker wire) to a potentiometer. Then one must write a tricky machine language (the faster the program the better, the larger number of samples the the higher the frequency response of the voice: frequency=sample rate/2 or something like that. If you don't understand, ask Jim Dahlberg) program to retrieve the voltages and store them in

memory for later processing.(Run through a sound channel in volume output only mode, in effect turning the sound channel to a crude digital to analog converter [usually 4 bit resolution but is possible to have 5 bit]) I tried this, but I gave up.
The other possibility, as discussed in the artical I mentioned last month, uses voices stored on cassette, and then reads them into memory for later use. This works for the Apple, But does the same method apply for the Atari (my question), and the answer is a resounding yes! Everything you need to know is in the technical manual. (I've had the silly thing for about a year now and am still learning things from it!) The 5 th bit of sksat-53775 (+ or - 2) will read either 0 or a 1, depending on the voltage. (this method doesn't allow as much waveform definition; the speaker can only be moved back and forth, wheras the previous method allowed up to 32 different speaker positions, a worth while trade off? You tell me.) All one needs to do to get some resemblance of real sound through the Atari is write a very simple machine language program (takes 5 minutes) that first: Poke 54018,52, then: LOOP LDA SKSAT, ROR A ,STA CONSOL -internal speaker, address in BASIC manual JMP LOOP. That's the basic idea. I did try this, I got the program running, and I stuck in a tape which contained some tunes, and I did hear a violin in between all the static. The idea needs to be refined. (The high notes, I think, are the ones that confuse the Atari, I think the hardware-POKEY, doesn't sample the cassette fast enough.)
If someone finds this absolutly all encompassing, and writes a program that emulates any instrument or human voice,

contact us. Where are my skis...
P.S. The magazine biliography will be delayed do to heavy snow. The ones to look at this month: -Creative Computing, loaded with stuff, Compute, of course, and the latest Micro had at least two programs in it.

FOR SALE-ATARI 800 48K,810 disk,820 40 column ptr,830 modem,850 interface All never used,$1700 call Manny Schule 715-357-3377

NEWS NOTES

Thanks to the efforts of the Twin Cities TRS-80 users group and Walt June of the Code Room in Eden Prairie, it should by now be common knowledge that you should not be paying sales tax on software. If any dealer charges you sales tax tell him to look at Minn. Sales and Use Regulation 610, and get your money back!

If you are going to by a printer you may want to check the classifieds in the newsletters of some of the other area user groups. These newsletters will be available to look at at TAIG meetings.

From ACE, Eugene, OR. Microsoft BASIC will cost $90 and take 20K of your memory. 20K!

From the Dec. TAIG meeting, held 1/3/82.
Software inventory forms were distributed. Please bring them to the Jan. meeting.

Fig-Forth interest group is being started, contact Steve Crowley 937-1001.

Computer Castle is offering a 32K board installed in your 400 for just the price of the board. This includes backing any remaining warrenty on the 400.

Price of the 800 is dropping to $899.

A Printer demo was requested.

TWINCITY
ATARI
INTEREST GROUP

6824 QUEEN AVE. SO.
RICHFIELD, MN 55423

TAIG Meeting Notice
Jan. 31, 1982, 7:00 PM
Minn. Fed. S&L
31 9th St.
Hopkins

A Cris Crawford video tutorial will be held at 6:00.